PROCEEDINGS OF THE SAWTOOTH SOFTWARE CONFERENCE

May 2020

CONJOINT MEETS AI

PETER KURZ
STEFAN BINNER
BMS MARKETING RESEARCH + STRATEGY

MOTIVATION FOR THIS PAPER

Artificial Intelligence (AI) and Artificial Neural Networks (ANNs) are the "talk of the town"! Most AI applications are located in the area of pattern recognition and big data. However, ANNs have also been used in the area of choice behavior in order to identify preference models (e.g., Bishop, 1995). Examples from the field of market research include models for price elasticities in FMCG and car ownership (e.g., Hensher & Ton, 2000; Mohammadian & Miller, 2002) and various other fields. The major advantage of ANNs is that they can efficiently recognize patterns in the data without being explicitly programmed as to where to look. This key feature of ANNs is called the Universal Approximation Theorem (Hornik et al., 1989) and describes their capability to approximate any Data Generating Process (DGP). However, despite the strong pragmatic appeal of ANNs, they have been criticized for being too much data-driven and theory-poor, in effect presenting the analyst with a black box model of the DGP.

As far as we know, past papers are mostly about how ANNs could be used to derive utility values from conjoint exercises (for example, Belyakov, 2019; Alwosheel, van Cranenburgh & Chorus, 2017). In these papers the conclusion is that hierarchical Bayes models perform as well as or better than ANNs for utility estimation. This is why we decided not to look into ANNs for estimation any more deeply, at least for the moment. New developments in the area of ANNs may change this situation in the future and it may become worthwhile to look into the utility estimation topic once more, but in the meantime HB does an excellent job!

The situation is different in the area of *experimental design* for Choice-Based Conjoint experiments, especially when alternative-specific designs are needed. We haven't found any literature so far that deals with ANNs and experimental designs. This is the reason why we want to further explore this topic. This is especially true because we encounter weaknesses in everyday work, mostly in the area of generating acceptable experimental designs for complex choice experiments. Calculating a statistically perfect design is an NP-hard problem.[1]

In day-to-day research work, client studies get more and more demanding, the numbers of attributes and levels are constantly increasing, and sample sizes get smaller and smaller. Therefore, in many cases it is not easy to find good experimental designs with the commonly used algorithms. Necessary restrictions and prohibitions on attribute levels (levels that can't be shown together) often push the limits in finding an appropriate design. Furthermore, most of the experimental designs used in day-to-day research were developed

1 NP-hardness (non-deterministic polynomial-time hardness), in computational complexity theory, is the defining property of a class of problems that are informally "at least as hard as the hardest problems in NP." This usually means that we can't solve the problem with brute-force algorithms because these would need almost endless time to run, except for trivially small cases.

to optimize *aggregate* models of choice behavior (MNL) and are not optimized to estimate heterogeneity in the context of hierarchical Bayes estimation.

Design Principles

Designs that are theoretically efficient usually have undesirable empirical properties: level balance, orthogonality, minimal overlap, and some degree of utility balance. A "perfect" design is completely uncorrelated, all levels appear equal times (preferably all two- and three-way combinations appear equally as well) and there is minimal (often meaning no) overlap between the attribute levels shown in one choice task. Such designs are superior from a statistical point of view, but are sometimes very strange to answer for respondents, because of implausible combinations. Choice tasks with a larger number of concepts, where most of the attributes have fewer levels than the number of concepts shown, often result in two nearly identical concepts due to the goal of reaching level balance. Think, for instance, of choice tasks where only the price attribute is varying and all other attribute levels are the same between the concepts. Other choice tasks might show one clearly superior ("dominant") product, so that the respondent's answer is obvious in advance. The variance of the model parameters estimated from a design depends on the actual parameter values, but most design algorithms assume equal preference weights in calculating efficiency (even with unknown preferences, attributes often have a natural ordering such as "mild," "moderate," and "severe," or a lower price which should be preferred over higher ones). This is the reason why classical designs often result in dominated pairs, where all the attribute levels of one alternative are better than the attribute levels of another alternative. Such choices provide no real preference information, even though they may be included in a theoretically efficient design.

Some researchers have concluded that a certain amount of utility balance (having alternatives about equally attractive overall) is needed, both to avoid odd choice tasks and to enhance efficiency (Huber & Zwerina, 1996). Rich Johnson, Joel Huber, and Bryan Orme (2005) conducted some practical experiments and showed that while utility balance is theoretically good, it usually doesn't result in improvements in empirical studies. This finding is attributed to the idea that utility balanced designs result in choice tasks with concepts that are more nearly equal in attractiveness and therefore make the choice tasks harder for respondents, causing more fatigue and/or random error. Also, creating utility balance requires prior information; if we are able to create perfect utility balance, we already know the answers!

We agree with both papers. On the one hand, no utility balance is bad for respondents, because they may see trivial choice tasks where the answer is clear to everyone. On the other hand, high utility balance may lead to too-difficult choice tasks. The net conclusion is, no utility balance at all and too high utility balance are both bad for empirical studies, but some degree of utility balance does help achieve better designs.

Another frequent concern when building statistically perfect designs is including implausible attribute combinations. An orthogonal design might include all possible combinations, including ones that are not possible in reality. Imagine an iPhone with an Android operating system. Eliminating these implausible combinations results in a design that is no longer orthogonal or level balanced.

Further problems constructing experimental designs often appear when line-pricing or a pricing system from the client has to be taken into account. Pricing systems could be such that if Coke increases in price, line-pricing for other soft drinks of the CocaCola company (e.g., Sprite) need to increase similarly Or, it might be a rule that a 10% price increase in sparkling water should mean a 30% increase for energy drinks. Such pre-conditions and relationships also lead to problems with level balance and orthogonality of the design.

How Do AI and ANNs Work?

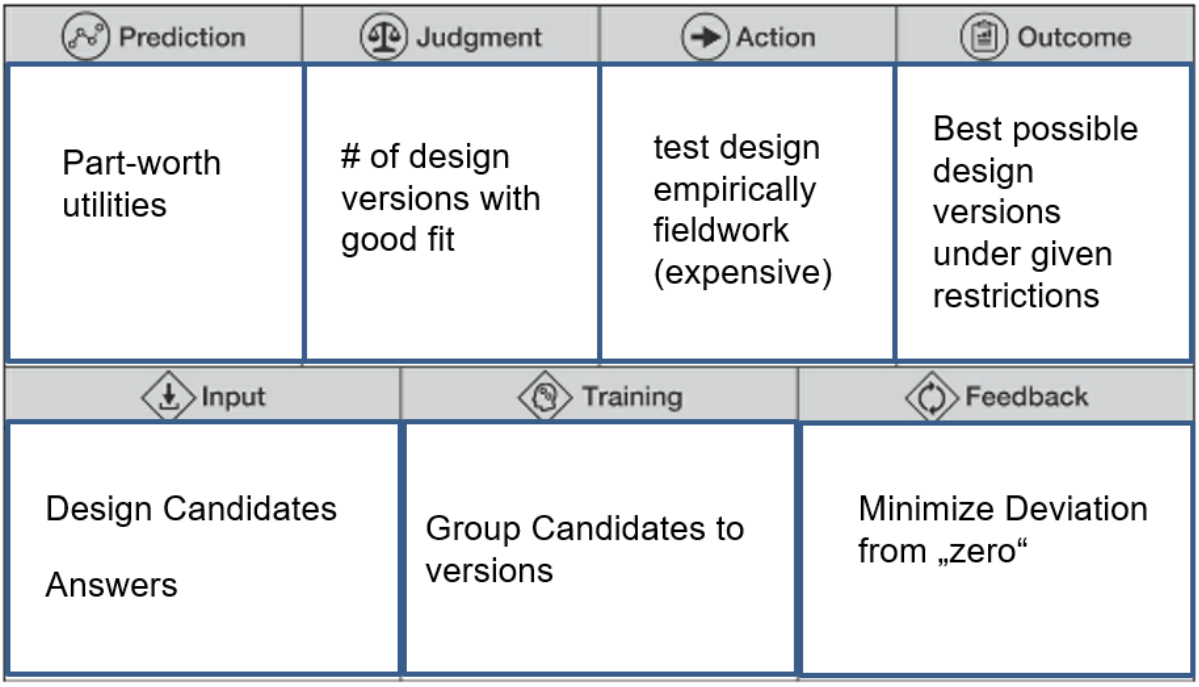
Taking the topics mentioned above into account, an AI-based design generation process has to find a nearly perfect experimental design and to minimize the statistical and measurement error. Such an AI approach should accommodate all practical needs like prohibitions, excluding implausible combinations and unavailable products, taking pricing systems into account, and including at least some utility balance to make the choice tasks more realistic. But, it must do this while giving up as little as possible of the desirable statistical properties (e.g., orthogonality, level-balance, overlap). From a statistical point of view, this means that we are looking for design versions in which purely random answers result in all estimated part-worth values being zero, or nearly so, and therefore an RMSE being as small as possible (standard errors close to zero). This sounds like more of a challenge than it actually is, because exactly such requirements are the strengths of ANNs.

What is the starting point for an ANN-based design approach?

- We know the answers to test (simply random answers).
- We know which prohibitions and pricing rules we must take into account.
- It is relatively easy to generate a large number of synthetic datasets to train our ANN's.
- We have a clear objective and "loss function": minimizing the standard errors of the part-worth coefficients.

Before building an AI tool it's necessary to decompose the workflow into separate tasks. Agrawal, Gans & Goldfarb (2018) suggest the use of an "AI canvas" (Figure 1) that helps to decompose the machine learning problem into tasks:

Figure 1: The AI Canvas



Source: Agrawal, A.; Gans, J.; Goldfarb, A. (2018): Prediction Machines.

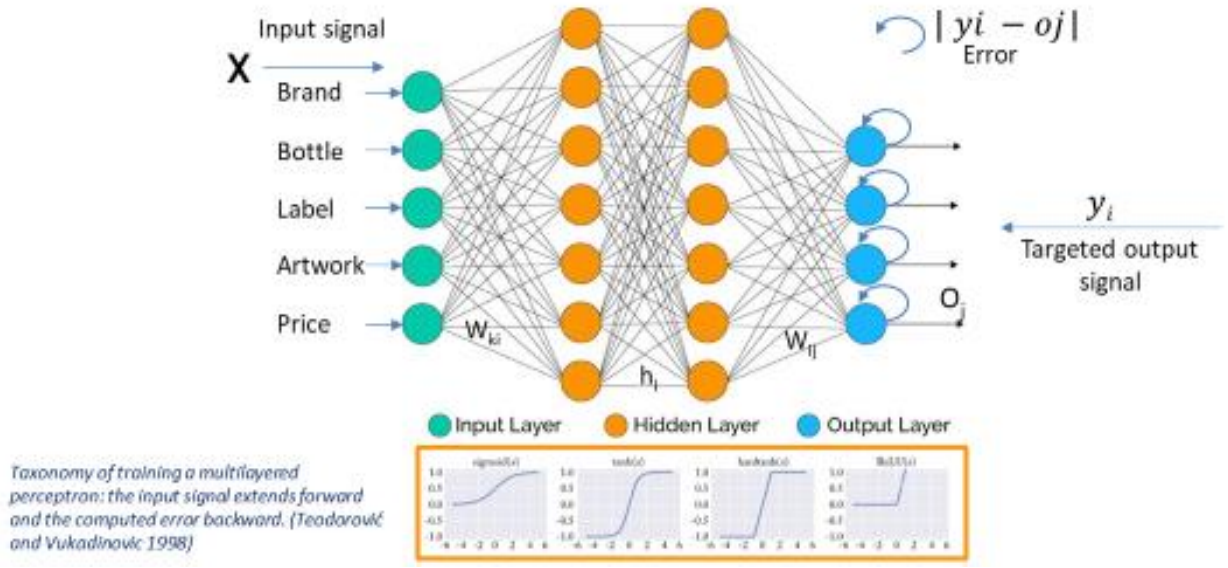
The first task is to **predict** the part-worths, so we need some process within our AI tool to estimate part-worth utilities. This could be done with the Softmax procedure implemented in the Keras[2] AI framework. Softmax is an ANN type that realizes multinominal logit calculations. The second task, **judgment**, determines how many good design versions we find in our input data, based on a maximum loss we will accept from the loss-function (standard error). The (third) **input** task involves generating the possible design versions. A design version is a complete set of choice tasks for one respondent. The candidate design versions can be produced either by fully random design generation or some specialized design algorithm. The (fourth) **training** task involves grouping design versions to make complete experimental designs that minimize the loss. The (fifth task) **feedback** loop is simply minimizing the loss function which means that we want to have part-worths with standard errors as small as possible. Information on the errors is sent back to the training task so it can try again to group the design versions into a different complete experimental design that results in a smaller loss. The (sixth task) **outcome** of this AI tool is the best possible experimental design (or at least, one very close to the best) under the given restrictions.[3] Finally, the (seventh) **action** task is to use the design in empirical studies and compare it to other designs used under similar conditions. Action is the most expensive step in the AI canvas. All the other steps only need computational power and can be done in the lab; action must be in the real world.

[2] Keras is a Python-based package for neural networks, accessible through R as well.

[3] What we call a “choice task” is the single question shown to a respondent. The choice task consists of a number of concepts. What we call a “version” or “design version” is the full set of choice tasks shown to one respondent, typically numbering 8 to 15. The complete experimental design consists of a defined number of versions, usually 20 to 40.

Since ANNs can be used to approximate any data generating process due to the universal approximation theorem, it should not be a problem to find an appropriate design. But, in order to find this solution ANNs need a large amount of data and a properly defined loss function. As shown above, we can easily fulfill both requirements: we can generate a large number of versions and answers and train the ANN to solve the problem by minimizing the loss function (small standard errors) and identify the optimal experimental design (combination of different versions) need for a complex choice experiment based on our input.

Figure 2: How ANNs Work



Source: **Teodorović, D., & Vukadinovic, K. (1998):** Traffic Control and Transport Planning: A Fuzzy Sets and Neural Networks Approach.

ANNs can be described as weighted directed graphs (Figure 2), where the nodes (colored circles) are the "neurons" and the connection lines between the neuron outputs and neuron inputs can be characterized by the targeted edges with weights (W_{ki}). Figure 2 is a simplified example of the real network we used. Our input is much more complex, one neuron for each attribute level * # of concepts * # number of choice tasks * # of versions needed for the experimental design + the synthetic answers for each choice task * number of synthetic respondents. As this is way too complex for an illustration, we show only one neuron for each attribute in Figure 2. The ANN collects the input signal from the external world in the form of a vector with binary information. These inputs are then mathematically defined by the notations x(n) for every n number of inputs, where n is the number of versions.

To make this more concrete, if we have 5 attributes with 3 levels each and 1 with 4 levels, for 19 total levels; 4 concepts per choice task; 12 tasks per version; and 20 versions desired for the complete design, our input layer would have 19 * 4 * 12 * 20 = 18,240 input neurons, each receiving a 0 or 1 input depending on whether a particular level applied to a particular concept in a particular task in a particular version. In addition, with 1,000 synthetic respondents, we would have 4 * 12 * 20 * 1000 input neurons indicating which of the four concepts were chosen in each task in each version by each simulated respondent.

Each of the inputs is then multiplied by its corresponding weights (these weights are determined by the training process used by the artificial neural networks to solve a specific problem). In common terms, these weights represent the intensity of the interconnection among neurons inside the ANN. All the weighted inputs are summed up inside the ANN (this could be seen as another artificial layer of neurons).

The "activation function" of each hidden or output neuron is some transformation of the weighted sum of the input values. It may be sigmoidal, or linear, or a step function, among other possibilities, as suggested by the bottom panel of Figure 2. The "softmax" neuron that implements logit choice is one specialized type of neuron. To understand the architecture of an ANN, we need to understand what components the neural network contains. A typical ANN includes a large number of artificial neurons which are units arranged in a series of layers. Let us take a closer look at the different layers available in an ANN:

Input Layer

The input layers contain those artificial neurons (units) which directly receive input from the outside world (input data), in our case, the design versions and answers.

Output Layer

The output layers consist of units that react to and reflect the information that is fed into the ANN. How closely their outputs (for us, part-worths) correspond to the desired ones (near-zero part-worths) reflects whether the ANN has learned any task well or not.

Hidden Layer

The hidden layers are located between the input layers and the output layers (not interacting with the outside of the ANN). The only job of a hidden layer is to convert the input into some meaningful form that the next layer can use in some way.

Most ANNs are completely interconnected, which means that all neurons in one layer are connected to all neurons in the next layer, leaving nothing unconnected. This allows a complete learning process. Learning occurs when the weights inside the ANN get updated after each new iteration.

In order to generate the input for training the ANNs we developed R code that generates large numbers of versions which are used as input for the first layer. The code generates versions based on a modula operation, by simply cycling thru all attribute-level combinations. In a second step the code deletes any tasks that fail any of our defined conditions: prohibitions, implausible combinations between concepts, violation of line-pricing, non-conformity with the needed pricing systems, violation of the desired amount of utility balance, and so on. To ensure that we produce enough possible versions, we use an oversampling strategy to bring in more versions with attribute-level combinations that are "relatively rare" because of the various restrictive conditions. We run these R scripts until we reach a large number of possible versions (typically a million or so) and then use those possible versions as input for the ANN.

The second part of the training data is the answers to the choice tasks. We simply generate random choices of concepts in each choice task. It is crucial to use a perfect

random distribution and avoid all positional effects. However, if the eventual real-world experiment will include a None option, it is important to define a realistic frequency of None answers and include them in the training data. Otherwise it is not possible to estimate the effect of None answers in the design, meaning that the efficiency will always be overestimated because of the assumption that all choice tasks will be answered on a forced-choice basis. Therefore, it is essential to have good estimates of real-world answering behavior, which is not always easy.

In order to train our ANNs we generated 1,000 synthetic respondents each answering all 15-20 choice tasks of a version. We replicated this for all of our 1,000,000 "design versions" that are used to build up the ANN. This procedure gave us sufficient information to estimate part-worth utilities at aggregate level and to have a large enough set of versions to stabilize the training of the ANNs.

Loss Function

A key task is to define the loss function. In the case of Choice-Based Conjoint exercises, it is simply the deviation from zero of the estimated part-worth utilities. The training process iteratively generates weights for the ANN layers which minimize the deviation from zero for the estimated part-worth utilities. As we want no single biased attribute levels, we minimize the deviation for all single attribute levels (parameters to estimate), so that we result in a loss-function combining the overall standard error and the standard error for each single parameter.

The proportion of the two components (overall and individual standard error) can either be a fixed ratio (as we have used in the example 70:30) or varied due to an optimization function (which introduces another layer). If we can get exact "zero"-values for all part-worth estimates, the design fulfills all design considerations. Deviations from zero cause or reflect weaknesses in at least one of the design considerations. Most often, orthogonality is violated (attribute levels are correlated). However, we know from the literature it is acceptable to give up some orthogonality in order to derive useful designs, especially when estimating multinomial logit models. Nevertheless, the better we meet the design considerations, the better our empirical study results.

For experimental designs where it is possible to fulfill all design considerations (it is often not possible, due to prohibitions that prevent orthogonality, for example), we can prove that ANNs are able to minimize the loss function to exactly zero values for all estimates.

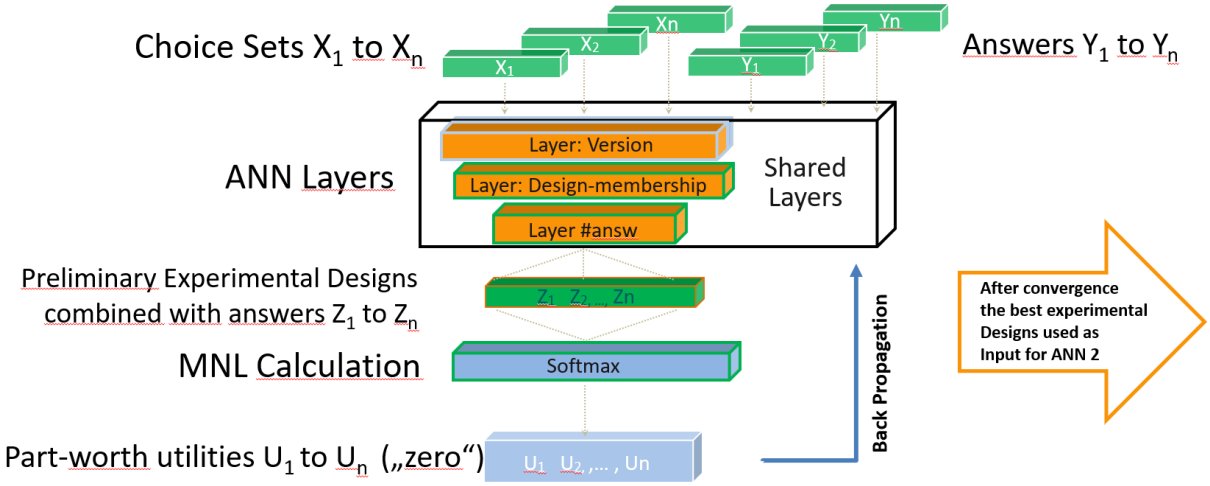
Training of the ANNs

Finally, we have to train the ANNs. Our experiment with different ANN strategies showed best results, if we train two ANNs: ANN1 for the experimental design that best fits at the aggregate level and ANN2 to optimize the resulting experimental designs for individual utility estimation.

For ANN1 we use the randomly generated versions and random answers to the choice tasks in them as input. Then we train the layers to find the optimal experimental design to estimate the aggregate MNL. This is done using a standard way of training ANNs called "backpropagation," an iterative process that adjusts the weights of the neurons to minimize

the loss. After convergence is reached, ANN1 is able to produce the best experimental design possible, for later field work. For the complete experimental design we use a sufficient number of versions, so that different respondents see different sets of choice tasks. From our experience we usually end up with 20 to 40 different versions for a complete experimental design that is optimal for fielding. Up to this point, the design is only tested on an aggregate level, meaning we have used aggregate logit models to define the loss function being minimized.

Figure 3: ANN1 — Design Optimized for *Aggregate* Utility Estimation



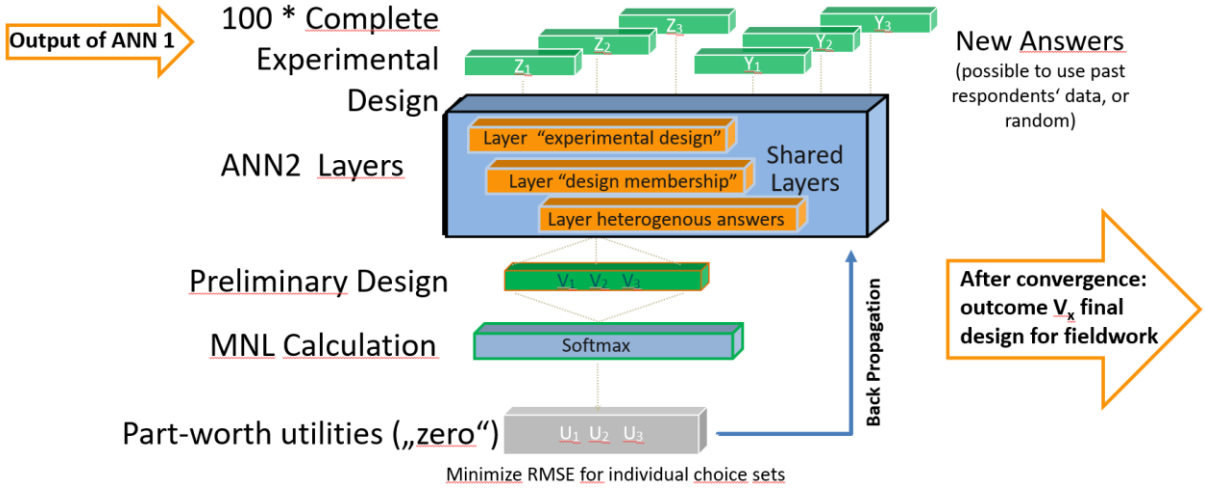
Flow of the first ANN: Input, Layers and Softmax Layer for MNL Calculation

ANN2 is trained to find the best experimental design for individual utility estimation (via HB) based on a sufficient number of possible experimental designs generated by ANN1. The input data are now complete experimental designs (the results from ANN1).[4] These designs are answered with new randomly generated answers. But at this stage we add some heterogeneity and eventually response error to the answers and train the ANN2 to minimize the deviation for each experimental design. To estimate on the individual level, the versions from each of the experimental designs are answered by 1,000 synthetic respondents, which are generated according to the assumed heterogeneity. For our training data we used 100 experimental designs, of 20 versions each, generated by ANN1 and answers to each of them by 1,000 synthetic respondents.

As one can see, in this phase we need assumptions about the real heterogeneity in the population. In future work we would like to use real data from past studies at this stage to generate synthetic respondents' answers, or at least generate answers based on knowledge of heterogeneity from past studies or markets.

[4] To implement the second step we uses a number of possible "nearly optimal"experimental designs from ANN1. Usually we find not just one perfect experimental design, but a group of possible experimental designs. If we end up with only one one perfect experimental design, we have to lower the restrictions of the loss-function a little and re-run to get more design options.

Figure 4: Design Optimized for *Individual* Utility Estimation



Flow of the second ANN: Input (results from ANN1), Layers and Softmax Layer for individual MNL Calculation

After we have trained the two ANNs, we are able to use the two networks to generate the best possible design versions for fieldwork. The ANNs have recovered the data-generating process which lies behind the design versions and the answers.

Bear in mind that the ANNs automatically structure the layers and calculate the weights, so there is no "programming" needed, the only information we used in the setup is the difference between versions and answers, and in ANN2 an indicator to which experimental design the versions belong. However, the resulting ANNs are "black boxes," so we don't have a chance to analyze them and see why some selections are superior to others and why we end up with those final design versions. This immediately shows the need of empirical testing on how good the results really are in reality!

Simulation Results

We ran hundreds of synthetic datasets to explore how well ANNs are able to generate ideal experimental designs when the underlying DGP (known utilities, Gumbel error as in standard MNL) is known to the analyst. We focus on standard criteria for good experimental designs like orthogonality, level balanced overlap, and utility balance (see Huber & Zwerina, 1996).

Table 1: Experiment Factors and Factor Levels

#	Factor	#Factor levels	Factor levels
1	# of parameters	6	20;40;60;80;120;160
2	# of prohibitions	6	0; 5, 10, 20, 40, 60,
3	Implausible products	5	0, 1, 5, 10, 20,
4	Line-Pricing	2	yes/no
5	Price System	2	yes/no
6	Utility balance	4	no, small, medium, high

There are 2.880 experimental conditions for the 6 experimental factors.
The number reduces slightly, cause some of the conditions could not be combined (for example 20 parameters with 120 prohibitions)

We used 6 experimental factors (Table 1) and varied them with 2 to 6 factor levels. For each of the combinations we generated experimental designs based on our two-stage ANN approach and answered them with synthetic respondents. As a comparison, we produced designs with the SAS Macros (Kuhfeld, 1996) with the same experimental factors and answered them with the same synthetic respondents. We chose SAS because it is possible to script the macros in SAS syntax to generate the over 2,000 designs automatically. In this experimental setting, we know the real utilities of each respondent and can show how well we reproduce the preferences of the artificial respondents.

To compare our different designs we used the root mean squared error (RMSE) over all part-worth utilities, the maximum standard error, and the range of the standard errors as criteria. Table 2 displays the results of this comparison for all experimental factors.

Table 2: Results from Simulation Study

Factor		# of parameters						# of prohibitions					
Level		20	40	60	80	120	160	0	5	10	20	40	60
RMSE	AI	**0.00**	**0.00**	0.03	0.18	0.28	0.36	**0.00**	**0.00**	0.10	0.13	0.15	0.26
	SAS*	**0.00**	**0.00**	0.03	0.20	0.32	0.38	**0.00**	**0.00**	0.10	0.15	0.17	0.29
Range	AI	**0.00**	0.05	0.09	0.12	0.19	0.22	**0.00**	0.01	0.08	0.11	0.20	0.31
	SAS*	**0.00**	0.14	0.17	0.24	0.38	0.73	**0.00**	0.01	0.14	0.16	0.24	0.68
Max. Std.Err.	AI	**0.00**	0.02	0.05	0.08	0.13	0.17	**0.00**	0.01	0.04	0.06	0.10	0.16
	SAS	**0.00**	0.05	0.09	0.15	0.22	0.39	**0.00**	0.01	0.09	0.11	0.18	0.41

Factor		# implausible products					Line Pricing		Price System		utility balance			
Level		0	1	5	10	20	no	yes	no	yes	no	small	medium	large
RMSE	AI	**0.00**	0.01	0.06	0.17	0.27	**0.00**	0.07	**0.00**	0.09	**0.00**	0.07	0.09	0.24
	SAS*	**0.00**	0.03	0.08	0.24	0.32	**0.00**	0.09			**0.00**			
Range	AI	**0.00**	0.06	0.13	0.20	0.25	**0.00**	0.07	**0.00**	0.27	**0.00**	0.07	0.18	0.21
	SAS*	**0.00**	0.09	0.13	0.25	0.35	**0.00**	0.13			**0.00**			
Max. Std.Err.	AI	**0.00**	0.05	0.07	0.10	0.16	**0.00**	0.05	**0.00**	0.19	**0.00**	0.04	0.10	0.21
	SAS*	**0.00**	0.07	0.08	0.16	0.22	**0.00**	0.84			**0.00**			

The differences between AI and SAS become larger the more complex the designs are: the more prohibitions and implausibilities that have to be taken into account, the better the ANN-based versions perform compared to the SAS designs. Furthermore, we can see that a high degree of utility balance harms the designs. This finding is in line with the literature: a small amount of utility balance is OK, but higher utility balance makes the designs worse. Note: in the case of synthetic data, the problem with utility balance is not respondent burden or error. With higher utility balance we simply produce larger deviations from orthogonality, and for complex designs with lots of prohibitions we also violate level balance. In the real world, it's likely that respondents answering behavior and higher burden, as Rich Johnson encountered in his studies, eventually add further to these errors.

The same pattern can be seen in hit rates and shares. The AI-based designs always have lower RMSE than the classical SAS-based designs. For the synthetic data, we can conclude that the black-box works and the ANNs are well-trained to produce appropriate designs and can handle much more complex design restrictions, compared to designs based on the classical algorithms (the SAS designs). The results for hit rates show that the ANN-based designs are better at capturing heterogeneity than the classical designs. This leads to the conclusion that the training effort for the second ANN pays off.

EMPIRICAL STUDIES

For empirical testing of ANN performance with real data we conducted two studies with split cell designs: one study about orange juice and one about chocolate bars. In each study, one cell used ANN-based designs, and the other used designs from the SAS macros.

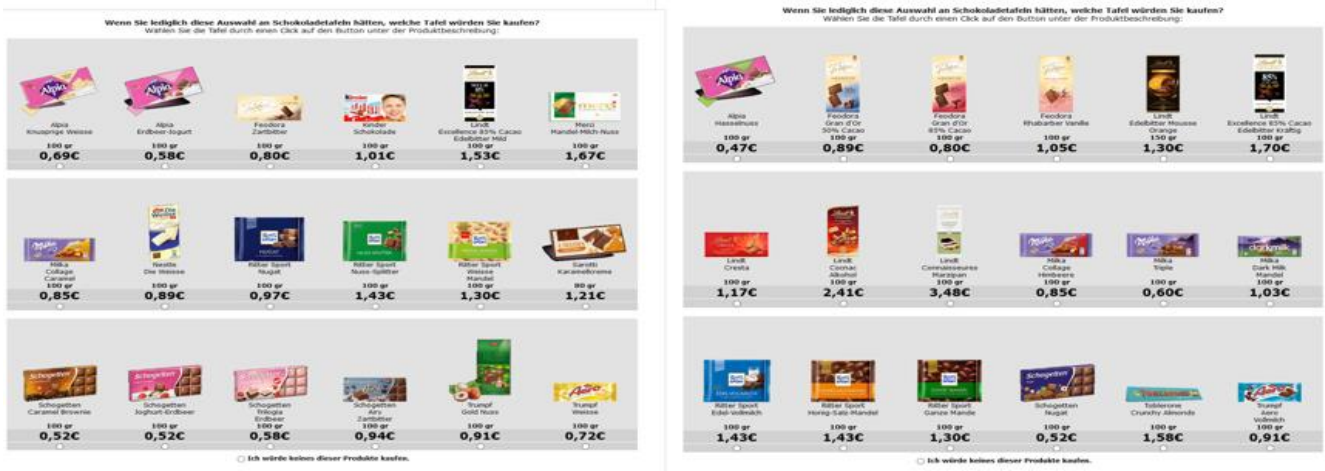
The orange juice study was conducted in Germany in December 2018 and January 2019 with respondents who bought orange juice in the last month, aged between 16 and 65 years. The sample was n=1,010 and n=1,005 for the two design splits. The objective of this study was to optimize the bottle type, the quality of juice, and the packaging artwork, and to get insights into the impact of quality labels like "organic" or "fair trade" (Figure 5). The design complexity comes from the problem that not all brands can produce all package types or all different qualities. Depending on the quality and pack type, not all prices could appear with all juices.

Figure 5: Choice Tasks from the Orange Juice Study

Orange Juice study Germany 2018/19; n=1010/1005

The second study, on chocolate bars, was also conducted in Germany, in August 2018 and July 2019 with two samples of 605 and 608 category buyers (at least once in the last month) and aged between 16 and 65 years. The objective of this exercise was to optimize the assortment and pricing, add new flavors to the market, and have an optimal differentiation from what the competitors offer. The design complexity was caused by the fact that not all brands can offer all flavors, some brands use very special ingredients that are branded and cannot be used by other brands, and some brands produce special chocolate variants with special cacao that cannot be shown with other brands. In addition, the assortment size is very different between the competitors and some ingredients are much more expensive than others (Figure 6).

Figure 6: Choice Tasks from the Chocolate Bar Study



Chocolate bar study Germany 2018/19; n=605/608

We expected that if we generated the experimental designs for the two studies with the ANN-based approach, it would be possible to improve level balance for one-way and two-way frequencies, as well as orthogonality. Furthermore, we used the ANN2 to add some utility balance for the price attribute, so that lower prices for the same type of chocolate bars are preferred. Due to the limited number of empirical test cells we weren't able to test whether a specific amount of level overlap and/or adding special tasks to estimate interaction effects would pay off. However, ANNs could easily address such criteria.

First, we compared the designs for the two studies, again with purely random answers. With this initial test we could show that the AI-based designs have a smaller RMSE, smaller standard deviation and smaller standard errors. Even though our ANNs were not trained to optimize D-efficiency, they beat the SAS designs in the area of their strength, D-efficiency, as well (Table 3).

Table 3: Comparison of the Two Design Approaches

Part-Worth-Utilities Estimated from Random Answers:

Study	Design	RMSE	Min	Max	Avg. Std. Err.	D-efficiency
1: Orange Juice	SAS	1.085	-0.087	0.097	0.032	82.53
	AI	0.953	0.000	0.013	0.005	83.45
2: Chocolate Bars	SAS	1.146	-0.078	0.125	0.048	83.01
	AI	0.741	-0.008	0.072	0.010	84.71

Orange Juice Germany 2018/19; n=1010/1005; Chocolate bar 2018/2019; n=605/608

In the orange juice study, we added additional feedback questions about respondents' experience when doing the exercise. In Table 4, we can see slight improvements in likeability, but even higher gains for the AI-based designs concerning the realism of the exercise compared to a real shopping trip. But more important is the dramatic increase in the number of choice tasks where respondents can make a choice, because they see a product that they would really like to buy. A real advantage of the ANN-based designs is the number of meaningful choice alternatives for respondents. Answers of respondents should be more realistic and meaningful if they really see products they want to buy and don't have to choose products they would never really consider buying in most of the tasks.

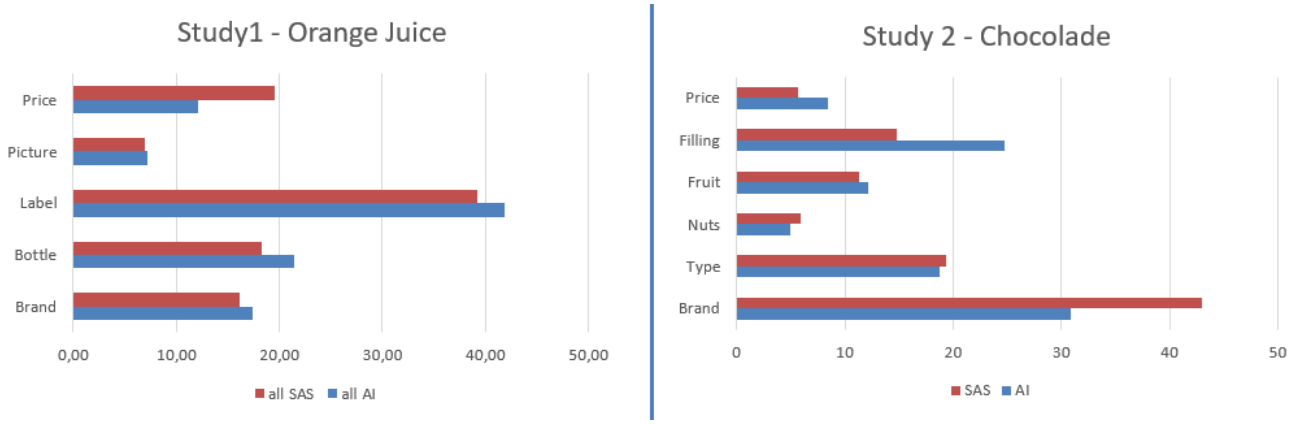
Table 4: Results from the Feedback Questions, “Orange Juice” Study

		SAS	AI
Overall Likeability	Extremely/very well done	19%	24%
Look & Feel	Like it	89%	90%
Realistic	Seems like a real shopping trip	10%	29%
	Comes close to a shopping trip	26%	38%
Do you see an orange juice in most of choice tasks that you would like to buy?	In nearly all	2%	12%
	In a large number	21%	**53%**
	In about 50%	31%	15%
	In less than half	**46%**	20%
		N= 1005	N=1010

Orange Juice study Germany 2018/19; n=1010/1005

A second finding is that the importances of the attributes are different between the two design strategies. In Figure 7, we can clearly see that, especially if prohibitions are used, design weaknesses influence the attribute importance. In the orange juice case, the prohibitions on labeling and different bottles affect the importance of the price parameter. In the chocolate bar study, the different amount of level balance, especially for the filling attribute, causes large differences in the importance of these attributes.

Figure 7: Attribute Importances from the Two Studies by the Two Design Strategies



Orange Juice Germany 2018/19; n=1010/1005; Chocolate bar 2018/2019; n=605/608

The comparison of in-sample hit rates is based on 6 selected random tasks (estimating the utilities 6 times by leaving one single random task out in each estimation). The result clearly showed that the hit rates on all holdout tasks are always better with the ANN-based

versions (Table 5). The better handling of prohibitions and implausible combinations, which results in better level balance, seems to pay off.

Table 5: Comparison of Within-Sample Hit Rates by Study and Design

	Within-sample hitrates			
	Study 1: Orange Juice		Study 2: Chocolate Bars	
	AI	SAS	AI	SAS
Random Holdout 1	68.9%	59.8%	69.3%	59.8%
Random Holdout 2	56.4%	54.8%	67.5%	51.2%
Random Holdout 3	58.5%	51.9%	56.5%	51.2%
Random Holdout 4	55.4%	51.7%	57.3%	52.2%
Random Holdout 5	57.9%	54.4%	60.5%	44.4%
Random Holdout 6	59.6%	52.6%	64.2%	48.0%

Orange Juice Germany 2018/19; n=1010/1005; Chocolate bar 2018/2019; n=605/608

Looking deeper into level balance, we see that the deviation of one-way frequencies is larger for the SAS-based designs. Table 6 shows that the SAS-based designs always have a larger difference in the one-way frequencies of levels shown in one design version to the respondents. The superior level balance is one reason why AI designs are always a little better.

Table 6: Selected One-Way Frequencies from the Chocolate Bar Study

Selected levels of Attribute "Brand":

Brand shown:	SAS	AI
Alpia	7-12	2-9
Lindt	6-12	1-8
Milka	19-26	6-15
Ritter	12-19	3-13
Tobler	3-6	1-6

Attribute "Chocolate Type":

Type shown:	SAS	AI
Milk	80-91	34-51
Dark	18-26	6-16
Black	5-11	1-8
White	4-9	1-6

Attribute "Contains Fruit"

Fruit:	SAS	AI
Fruit	3-8	1-6
Berry	6-10	1-7
Grape	4-8	1-6
None	93-105	43-54

Selected Level of Attribute "Filling"

Filling:	SAS	AI
coffee	3-5	1-4
Nougat	2-8	1-7
Marzipan	2-11	1-6
Caramel	8-20	1-9

Chocolate bar study 2018/2019; n=605/608

Much more impressive are the differences in the two-way frequency tables (Table 7). For example, the attribute level "milk chocolate" is highly correlated with brands in the SAS design (only 15 views for Lindt, 554 for Milka). The AI-based designs do a much better job of achieving pairwise-level balance. Prohibitions of combinations didn't affect the ANN design generation nearly as much as they affect the classical approaches.

Table 7: Two-Way Frequencies of Selected Attributes from the Chocolate Bar Study

Example: Selected level of "Brand" by Attribute "Chocolate Type:

Brand shown	Type shown							
	Milk		Dark		Black		White	
	SAS	AI	SAS	AI	SAS	AI	SAS	AI
Alpia	553	546					54	80
Lindt	15	239	409	237	183	124	5	21
Milka	554	554	61	117			2	43
Ritter	484	503	5	62	2	14	116	47
Tobler	188	351	242	153			177	122

Chocolate bar study 2018/2019; n=605/608

For the chocolate bar study, we have real market data available and therefore we could do some out-of-sample predictions. We compared the base case share of choice for both designs with the market shares from the German chocolate market (Table 8).

Table 8: Error Measures[5] for Share of Choice for the Two Empirical Studies

Study		MAE*	MSE*	RMSE*	MAPE*	RAE*
Study 1: Orange Juice	SAS	0.781	0.963	0.984	3.312	0.329
	AI	0.642	0.524	0.726	2.491	0.242
Study 2: Chocolade Bars	SAS	1.194	0.013	0.116	1.291	0.161
	AI	1.069	0.002	0.005	0.831	0.101

Orange Juice Germany 2018/19; n=1010/1005; Chocolate bar 2018/2019; n=605/608

All error measures are slightly better for the results based on the AI designs. Measures like RMSE or MAPE that penalize larger deviations more than smaller ones especially show that the AI designs are superior to the classical ones. Both in-sample and out-of-sample results are better for the ANN-based design versions.

[5] All error measures are better when lower. MAE is mean absolute error. MSE is mean squared error (a variance-like measure that weights larger prediction errors more heavily than small ones). RMSE is root mean squared error, the square root of MSE and standard deviation-like. RMSE penalizes larger prediction errors more strongly but finally reports the average prediction error in the dimension of the original measurement units. MAPE is mean absolute percentage error (error as a percentage of the true value). RAE is relative absolute error, the ratio of an RMSE-like measure to the same measure for a naïve equal-probability model.

FINDINGS

As expected, the AI-based designs fit perfectly to the assumptions we included in our candidate generating process. In other words, the AI recovered the DGP. Prohibitions, implausible products, ordered attributes, assumptions about utilities, and pricing strategies were better accommodated better in the AI design versions. Overall, RMSEs, SDs, part-worth estimates, one-way and two-way frequencies, as well as D-efficiency are all closer to optimal designs in the AI-based approach than in the SAS-developed designs.

Adding a large amount of utility balance harmed the designs. Rich Johnson was right, we do not always gain an advantage by forcing utility balance in empirical data.

Small to medium amounts of utility balance (especially for ordinal attributes and price) results in better designs and showed no disadvantages on respondent burden and fatigue answering behavior.

From our two empirical studies we can conclude that there are differences in the estimated part-worth utility depending on the design generation technique. AI-based design techniques appear to deliver more stable and better results, although two empirical studies are not enough to conclude that they are always superior. We can see in our two examples that AI-based designs were superior in all tested measures. We see more valid results in case of attribute importance (proof of which is only possible in simulation studies) and more face validity, at least, for attribute importance in the empirical studies. Shares of choice (both simulated and in empirical studies) are closer to reality and have smaller errors. In our empirical studies we saw that the within-sample hit rates were higher when using AI-based choice tasks. In our second empirical study we saw that out-of-sample error between real market and predicted-share was slightly reduced with the ANN-based designs.[6]

In general, we can conclude that the Universal Approximation Theorem from ANN theory can be applied in the context of experimental design for Choice-Based Conjoint studies, meaning the ANNs are able to identify the DGP and come up with stable design versions. Although predictions generated by deep learning and many other AI technologies appear to be created from a black box, we can say that in our context the black box works well.

Most studies conducted in the marketing research community are still based on design algorithms which were developed for aggregate models. We have presented here a new technique which is able to generate optimized experimental designs for individual-level estimates.

FUTURE WORK

We need further investigations based on more empirical studies with out-of-sample data. We also need more work on what happens if we include wrong assumptions (such as too much utility balance)! So far, we can only suggest being careful when defining your assumptions for the candidates. We don't really know what happens if the assumptions about utility balance are wrong.

[6] No data on real market share was available for the first empirical study.

A major potential step forward is the possibility of including "revealed preference" (past respondents' data) in the training of the ANNs to derive better designs for individual parameter estimates. If we have information from past choice models or panel data, we can train the second ANN with answers that have the same heterogeneity, same class memberships, and same utility structure as the revealed preference data. The trained ANN could then produce optimal deigns for fitting exactly to the actual respondent heterogeneity. ANN-generated designs based on past data could result in better input for pseudo-individual utility estimation (ability to capture more heterogeneity) by optimizing these designs for individual-level estimates.

If no past data are available, we can try, instead of using simple random answers, incorporating different answering routines in the computation, to come closer to real respondent answering behavior and the structure of real datasets, for use in the training phase of the ANNs.

As the proportion of None answers influences the results, we need further investigation of what happens if we assume a too large or too small None share in the training phase. Additional backpropagation loops to investigate the influence of None answers on the design during the training should be implemented and tested.

Peter Kurz

Stefan Binner

REFERENCES

Agrawal, A.; Gans, J.; Goldfarb, A. (2018): *Prediction Machines*, Harvard Business Press, Boston.

Allenby, G.M.; Rossi, P.E. (2006): Hierarchical Bayes Models, in: Grover, R.; Vriens, M. (Eds.): *The Handbook of Marketing Research: Uses, Misuses, and Future Advances*, 418–440, SAGE Publications Inc., Thousand Oaks.

Alwosheel, A.; van Cranenburgh, S.; Chorus, C. (2017): Artificial neural networks as a means to accommodate decision rules in choice models, Presentation at the 5th International Choice Modelling Conference 2017.

Anderson, D.A.; Wiley, J.B. (1992): Efficient choice set designs for estimating cross-effects models, *Marketing Letters*, 3(4), 357–70.

Belyakov, D. (2018): Applying machine learning to CBC data, SKIM/Sawtooth Software European Conference Paris, April 2019.

Belyakov, D. (2017): kNNLogit Ensembles for CBC Studies, AMA ART Forum 2017.

Bishop, C. M. (1995): *Neural networks for pattern recognition*, Oxford University Press.

Bunch, D.S.; Louviere, J.J.; Anderson, D.A. (1994): A Comparison of Experimental Design Strategies for Multinomial Logit Models: The Case of Generic Attributes, working paper, Graduate school of Management, University of California at Davis.

Burgess, L.; Street, D.J. (2005): Optimal designs for choice experiments with asymmetric attributes, *Journal of Statistical Planning and Inference*, 134(1), 288–301.

Chorus, C.G. (2010): A new model of Random Regret Minimization, *European Journal of Transport and Infrastructure Research*, 10(2), 181–196.

Chorus, C.G. (2014): Capturing alternative decision rules in travel choice models: A critical discussion, Chapter 13 (pages 290–310) in Hess, S. & Daly, A. (Eds.) *Handbook of Choice Modelling*, Edward Elgar Pub.

Hensher, D. A.; Ton, T. T. (2000): A comparison of the predictive potential of artificial neural networks and nested logit models for commuter mode choice, *Transportation Research Part E*, 36(3), 155–172.

Hess, S.; Stathopoulos, A.; Daly, A. (2012): Allowing for heterogeneous decision rules in discrete choice models: an approach and four case studies, *Transportation*, 39(3), 565–591.

Hornik, K.,; Stinchcombe, M.; White, H. (1989): Multilayer feedforward networks are universal approximators, *Neural Networks*, 2(5), 359–366.

Huber, J.; Zwerina, K. (1996): The importance of utility balance and efficient choice designs, *Journal of Marketing Research*, 33(3), 307–17.

Johnson, R.M.; Huber, J.; Orme, B. (2005): A Second Test of Adaptive Choice-Based Conjoint Analysis (The Surprising Robustness of Standard CBC Designs). Proceedings of the 2004 Sawtooth Software Conference , Sawtooth Software Inc. Sequim WA.

Kuhfeld, W.F.; Tobias, R.D.; Garratt, M. (1994): Efficient experimental design with marketing research applications, *Journal of Marketing Research*, 43(3), 409–19.

Kuhfeld, W.F.: (2000): *Conjoint Analysis Examples*, SAS Institute TS-650F, Cary (NC).

Kuhfled, W.F. (1996): Marketing Research Methods in the SAS System, SAS Institute, Cary (NC).

Lazari, A.G.; Anderson, D.A. (1994): Design of Discrete Choice Experiments for Estimating Both Attribute and Availability Cross Effects, *Journal of Marketing Research* 31; 375–83.

Leong, W.; Hensher, D. A. (2012): Embedding decision heuristics in discrete choice models: A review, *Transport Reviews*, 32(3), 313–331.

Louviere, J.J.; Woodworth, G. (1983): Design and analysis of simulated consumer choice or allocation experiments: an approach based on aggregate data, *Journal of Marketing Research*, 20(4), 350–67.

McFadden, D. (2001): Economic choices, *The American Economic Review*, 91(3), 351–378.

Mohammadian, A.; Miller, E. (2002): Nested logit models and artificial neural networks for predicting household automobile choices: Comparison of performance, *Transportation Research Record*, (1807), 92–100.

Rao, Vithala R. (2014): Applied Conjoint Analysis. Springer Heidelberg, New York.

Teodorović, D.; Vukadinovic, K. **(1998):** *Traffic Control and Transport Planning: A Fuzzy Sets and Neural Networks Approach*. Kluwer Academic Publishers, 1st edition.

van Cranenburgh, S.; Guevara, C. A.; Chorus, C. G. (2015): New insights on random regret minimization models, *Transportation Research Part A*, 74, 91–109.

Vapnik, V. N.; Chervonenkis, A. Y. (2015): On the uniform convergence of relative frequencies of events to their probabilities. In: Vovk,V.; Papadopolos,H.; Gammerman, A. (eds) *Measures of Complexity* (pp. 11–30), Springer, Cham.